WEST Search History

Hide Items | Restore | Clear | Cancel

DATE: Sunday, September 05, 2004

Hide?	Set Name	Query	Hit Count
		DB=USPT; THES=ASSIGNEE; PLUR=YES; OP=OR	
☐	L14	L12 and (xml or html)	5
☐	L13	L12 and (ml or html)	5
☐	L12	L6 and database	6
☐	L11	L3 and database	1
☐	L10	L8 and database	0
☐	L9	L1 and database	9
		DB=EPAB,JPAB,DWPI,TDBD; THES=ASSIGNEE; PLUR=YES; OP=OR	
☐	L8	(pars$ with order$) same (tag$ with (charge or cost$ or value or amount))	4
☐	L7	=20010109	0
		DB=USPT; THES=ASSIGNEE; PLUR=YES; OP=OR	
☐	L6	l1 and (rule or ruling)	6
☐	L5	L4 and hierarch$	7
☐	L4	l1 not L3	10
☐	L3	L1 and (contract$ and tag$)	1
☐	L2	L1 and (contract$ same tag$)	0
☐	L1	=20010109	11

END OF SEARCH HISTORY

h e b b cg b chh e f c e c g e

L5: Entry 4 of 7 File: USPT Oct 30, 2001

US-PAT-NO: 6311327
DOCUMENT-IDENTIFIER: US 6311327 B1

TITLE: Method and apparatus for analyzing software in a language-independent manner

DATE-ISSUED: October 30, 2001

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
O'Brien; Stephen Caine	Tigard	OR		
Maxwell, III; Sidney R.	Bothell	WA		

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
Applied Microsystems Corp.	Redmond	WA			02

APPL-NO: 09/ 250126 [PALM]
DATE FILED: February 12, 1999

PARENT-CASE:
CROSS-REFERENCE TO RELATED APPLICATION This application is a continuation-in-part of U.S. patent application Ser. No. 09/035,308, filed Mar. 2, 1998, now U.S. Pat. No. 6,161,200.

INT-CL: [07] G06 F 9/44

US-CL-ISSUED: 717/4; 717/8, 714/35, 714/38, 714/45
US-CL-CURRENT: 717/114; 714/35, 714/38, 714/45, 717/128, 717/130, 717/131, 717/140

FIELD-OF-SEARCH: 717/4, 717/8, 714/35, 714/38, 714/39, 714/45, 712/227

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

	PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
☐	5265254	November 1993	Blasciak et al.	717/4
☐	5408650	April 1995	Arsenault	395/575
☐	5450586	September 1995	Kuzara et al.	717/4
☐	5581695	December 1996	Knoke et al.	714/28

h e b b g e e e f c e e ge

☐	5615332	March 1997	Yamamoto	714/38
☐	5737520	April 1998	Gronlund et al.	714/39
☐	5748878	May 1998	Rees et al.	714/38
☐	5771345	June 1998	Tallman et al.	714/30
☐	5903759	May 1999	Sun et al.	717/4
☐	5954826	September 1999	Herman et al.	714/46
☐	6016557	January 2000	Kasprzyk et al.	714/38
☐	6047390	April 2000	Butt et al.	714/43
☐	6094730	July 2000	Lopez et al.	714/28
☐	6106571	August 2000	Maxwell	717/4
☐	6161200	December 2000	Rees et al.	714/38

FOREIGN PATENT DOCUMENTS

FOREIGN-PAT-NO	PUBN-DATE	COUNTRY	US-CL
0 567 722 A2	November 1993	EP	

OTHER PUBLICATIONS

Crooks, Roger "Embedded RISC .mu.Ps Present New Debugging Challenges," EDN, 39 (16):105-112, Aug. 4, 1994.

ART-UNIT: 212

PRIMARY-EXAMINER: Dam; Tuan Q.

ATTY-AGENT-FIRM: Seed IP Law Group PLLC

ABSTRACT:

A software analysis system for capturing tags generated by tag statements in instrumented source code. The software analysis system includes a probe that monitors the address and data bus of the target system. When a tag statement is executed in the target system, a tag is written to a predetermined location in the address space of the target. The tag contains a tag value that is indicative of the location in the source code of the tag statement generating the tag. By monitoring the predetermined address, the probe is able to capture tags as they are written on the data bus of the target system. The source code instrumenter includes a language-dependent parser and a language-independent analyzer that records tagging data in a symbol database. The software analysis system may reference the tagging data in the symbol database while testing instrumented source code. The software analysis system performs a variety of analysis functions in essentially real time, including code coverage, function and task execution times, memory allocation, call pairs, and program tracing.

34 Claims, 19 Drawing figures

Previous Doc Next Doc Go to Doc#

h e b b g e e e f c e e ge

A

L8: Entry 3 of 4 File: DWPI Jun 24, 2002

DERWENT-ACC-NO: 2002-528222
DERWENT-WEEK: 200267

TITLE: Data structure conversion for communications system by using definition language with data structure elements represented as bit strings

INVENTOR: NUUTTILA, P

PATENT-ASSIGNEE: NOKIA CORP (OYNO)

PRIORITY-DATA: 2000FI-0002720 (December 12, 2000)

PATENT-FAMILY:

PUB-NO	PUB-DATE	LANGUAGE	PAGES	MAIN-IPC
AU 200217188 A	June 24, 2002		000	G06F005/00
WO 200248856 A1	June 20, 2002	E	061	G06F005/00
FI 200002720 A	June 13, 2002		000	H04L000/00

DESIGNATED-STATES: AE AG AL AM AT AU AZ BA BB BG BR BY BZ CA CH CN CO CR CU CZ DE DK DM DZ EC EE ES FI GB GD GE GH GM HR HU ID IL IN IS JP KE KG KP KR KZ LC LK LR LS LT LU LV MA MD MG MK MN MW MX MZ NO NZ PH PL PT RO RU SD SE SG SI SK SL TJ TM TR TT TZ UA UG US UZ VN YU ZA ZW AT BE CH CY DE DK EA ES FI FR GB GH GM GR IE IT KE LS LU MC MW MZ NL OA PT SD SE SL SZ TR TZ UG ZM ZW

APPLICATION-DATA:

PUB-NO	APPL-DATE	APPL-NO	DESCRIPTOR
AU 200217188A	December 11, 2001	2002AU-0017188	
AU 200217188A		WO 200248856	Based on
WO 200248856A1	December 11, 2001	2001WO-FI01083	
FI 200002720A	December 12, 2000	2000FI-0002720	

INT-CL (IPC): G06 F 5/00; H04 L 0/00; H04 L 12/66

ABSTRACTED-PUB-NO: WO 200248856A
BASIC-ABSTRACT:

NOVELTY - Method consists in browsing the source bit string, identifying each data structure element (DSE) in it without parsing and handling the DSEs in the same order as they are in the data structure by decoding and encoding the decoded DSE directly into the target bit string (TBS). Each function is via a function call generated using coding rules and definitions in accordance with ASN.1 or IDL. The

h e b b g e e e f c e e ge

source bit string in BER is converted to the TBS in CDR. The DSE has a tag and a value and the utility set used for the steps is created using class libraries containing classes and a definition file.

DETAILED DESCRIPTION - There are INDEPENDENT CLAIMS for:

(1) A utility for performing data structure conversions in a communication system

(2) A utility set compiler

(3) A system for performing data structure conversions in a communications system

USE - Method is for data conversion between INAP/MAP messages used in the SS7 network and other messages used in distributed systems as applied to CORBA.

ADVANTAGE - Method enables SS7 network users to use CORBA services and vice versa faster with maximized automation of bridge maintenance.

DESCRIPTION OF DRAWING(S) - The figure shows IWU as a bridge between the SS7 network and CORBA.

ABSTRACTED-PUB-NO: WO 200248856A
EQUIVALENT-ABSTRACTS:

CHOSEN-DRAWING: Dwg.2/6

DERWENT-CLASS: T01 W01
EPI-CODES: T01-N02A1; T01-N02B; T01-S03; W01-A06E; W01-A06F2; W01-A06F3;

Previous Doc Next Doc Go to Doc#

A

L8: Entry 4 of 4 File: DWPI Jul 17, 2001

DERWENT-ACC-NO: 2001-624238
DERWENT-WEEK: 200172

TITLE: Function calling and execution method for computer program implementation wherein functions are mapped to function modules and function calls are parsed for parameters or parameter tags before execution

INVENTOR: FISHER, K L; HATCH, C A ; MULLINS, T W ; THURGOOD, B W ; VINCENT, R J

PATENT-ASSIGNEE: NOVELL INC (NOVEN)

PRIORITY-DATA: 1997US-0806505 (February 24, 1997)

PATENT-FAMILY:

	PUB-NO	PUB-DATE	LANGUAGE	PAGES	MAIN-IPC
☐	US 6263376 B1	July 17, 2001		015	G06F009/00

APPLICATION-DATA:

PUB-NO	APPL-DATE	APPL-NO	DESCRIPTOR
US 6263376B1	February 24, 1997	1997US-0806505	

INT-CL (IPC): G06 F 9/00; G06 F 9/45; G06 F 9/46

RELATED-ACC-NO: 1999-153261

ABSTRACTED-PUB-NO: US 6263376B
BASIC-ABSTRACT:

NOVELTY - A script command is first parsed to map to the required function module. The parameter tags and associated values are then assigned to function variables. Unassigned function variables are set to default values. Once the function call is complete, the next function call in the script is loaded. If a different function module is required, a search and load operation takes place.

DETAILED DESCRIPTION - An INDEPENDENT CLAIM is also included for a computer system and program using the generic binding interpreter.

USE - Function handling in object orientated programming languages.

ADVANTAGE - As the parser processes the parameter tags and assigns the parameter values, the function will work regardless of the tag order, avoiding the need for rigid parameter structures. It also allows only required variables to be included,

h e b b g e e e f c e e ge

not all variables available in the function, as the parser inserts default values for unused variables. This shortens the parameter lists to only required values making functional debug easier.

DESCRIPTION OF DRAWING(S) - The drawing shows a flow chart of the function call process including the module search operation.

ABSTRACTED-PUB-NO: US 6263376B
EQUIVALENT-ABSTRACTS:

CHOSEN-DRAWING: Dwg.3/5

DERWENT-CLASS: T01
EPI-CODES: T01-F02; T01-F04; T01-F05A; T01-F07; T01-J20C; T01-S03;

Previous Doc Next Doc Go to Doc#

Hit List

Clear | Generate Collection | Print | Fwd Refs | Bkwd Refs | Generate OACS

Search Results - Record(s) 1 through 7 of 7 returned.

1. Document ID: US 6763499 B1

L5: Entry 1 of 7 File: USPT Jul 13, 2004

US-PAT-NO: 6763499
DOCUMENT-IDENTIFIER: US 6763499 B1

TITLE: Methods and apparatus for parsing extensible markup language (XML) data streams

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Draw De

2. Document ID: US 6519617 B1

L5: Entry 2 of 7 File: USPT Feb 11, 2003

US-PAT-NO: 6519617
DOCUMENT-IDENTIFIER: US 6519617 B1

TITLE: Automated creation of an XML dialect and dynamic generation of a corresponding DTD

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Draw De

3. Document ID: US 6476833 B1

L5: Entry 3 of 7 File: USPT Nov 5, 2002

US-PAT-NO: 6476833
DOCUMENT-IDENTIFIER: US 6476833 B1

TITLE: Method and apparatus for controlling browser functionality in the context of an application

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Draw De

4. Document ID: US 6311327 B1

L5: Entry 4 of 7 File: USPT Oct 30, 2001

US-PAT-NO: 6311327
DOCUMENT-IDENTIFIER: US 6311327 B1

h e b b g e e e f e ef b e

TITLE: Method and apparatus for analyzing software in a language-independent manner

Full | Title | Citation | Front | Review | Classification | Date | Reference | | | Claims | KWIC | Draw De

☐ 5. Document ID: US 6279015 B1

L5: Entry 5 of 7 File: USPT Aug 21, 2001

US-PAT-NO: 6279015
DOCUMENT-IDENTIFIER: US 6279015 B1

TITLE: Method and apparatus for providing a graphical user interface for creating and editing a mapping of a first structural description to a second structural description

Full | Title | Citation | Front | Review | Classification | Date | Reference | | | Claims | KWIC | Draw De

☐ 6. Document ID: US 6085196 A

L5: Entry 6 of 7 File: USPT Jul 4, 2000

US-PAT-NO: 6085196
DOCUMENT-IDENTIFIER: US 6085196 A

TITLE: Object-oriented system and computer program product for mapping structured information to different structured information

Full | Title | Citation | Front | Review | Classification | Date | Reference | | | Claims | KWIC | Draw De

☐ 7. Document ID: US 6009436 A

L5: Entry 7 of 7 File: USPT Dec 28, 1999

US-PAT-NO: 6009436
DOCUMENT-IDENTIFIER: US 6009436 A

TITLE: Method and apparatus for mapping structured information to different structured information

Full | Title | Citation | Front | Review | Classification | Date | Reference | | | Claims | KWIC | Draw De

Clear | Generate Collection | Print | Fwd Refs | Bkwd Refs | Generate OACS

Terms	Documents
L4 and hierarch$	7

h e b b g e e e f e ef b e

9/757,227

WEST Search History

Hide Items | Restore | Clear | Cancel

DATE: Sunday, September 05, 2004

Hide?	Set Name	Query	Hit Count
		DB=USPT; THES=ASSIGNEE; PLUR=YES; OP=OR	
☐	L12	L6 and database	6
☐	L11	L3 and database	1
☐	L10	L8 and database	0
☐	L9	L1 and database	9
		DB=EPAB,JPAB,DWPI,TDBD; THES=ASSIGNEE; PLUR=YES; OP=OR	
☐	L8	(pars$ with order$) same (tag$ with (charge or cost$ or value or amount))	4
☐	L7	=20010109	0
		DB=USPT; THES=ASSIGNEE; PLUR=YES; OP=OR	
☐	L6	l1 and (rule or ruling)	6
☐	L5	L4 and hierarch$	7
☐	L4	l1 not L3	10
☐	L3	L1 and (contract$ and tag$)	1
☐	L2	L1 and (contract$ same tag$)	0
☐	L1	=20010109	11

END OF SEARCH HISTORY

h e b b cg b chh e f c e c g e

9/ 757227

Y-

L5: Entry 2 of 7 File: USPT Feb 11, 2003

US-PAT-NO: 6519617
DOCUMENT-IDENTIFIER: US 6519617 B1

TITLE: Automated creation of an XML dialect and dynamic generation of a corresponding DTD

DATE-ISSUED: February 11, 2003

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Wanderski; Michael C.	Durham	NC		
Wesley; Ajamu A.	Raleigh	NC		

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
International Business Machines Corporation	Armonk	NY			02

APPL-NO: 09/ 288341 [PALM]
DATE FILED: April 8, 1999

PARENT-CASE:
RELATED INVENTIONS IBM application Ser. No. 09/285,755 entitled "Applying Transformations to Legacy HTML Documents to Create Well-Formed Output" and Ser. No. 09/288,838 entitled "Achieving Complex Transformations with Dynamic Style Sheet Coalescing", filed concurrently herewith on Apr. 8, 1999.

INT-CL: [07] G06 F 15/00

US-CL-ISSUED: 707/513; 707/501.1, 707/500
US-CL-CURRENT: 715/513; 715/500, 715/501.1

FIELD-OF-SEARCH: 707/513, 707/501, 707/500, 707/511, 707/523, 707/501.1

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

	PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
☐	6006242	December 1999	Poole et al.	707/531
☐	6014680	January 2000	Sato et al.	707/513
	6023714	February 2000	Hill et al.	345/760

h e b b g e e e f c e e ge

☐	6061697	May 2000	Nakao	707/513
☐	6083276	July 2000	Davidson	717/1
☐	6226675	May 2001	Meltzer et al.	709/223
☐	6253366	June 2001	Mutschler, III	717/1

OTHER PUBLICATIONS

Spyglass Prims, "Concepts and application", copyright 1997 Spyglass, Inc., pp. 1-8.*
Brickmore et al., "Digestor: device-independent access to the World Wide Web", copyright 1997, pp. 1075-1082.*
Hill et al., "A Virtual Document Interpreter for Reuse of Information", 1998, pp. 1-2.*
"Spyglass Prim 1.0", copyright 1997, Spyglass, Inc., pp. 1-2.

ART-UNIT: 2176

PRIMARY-EXAMINER: Hong; Stephen S.

ASSISTANT-EXAMINER: Huynh; Thu V.

ATTY-AGENT-FIRM: Ray-Yarletts; Jeanine S. Doubet; Marcia L.

ABSTRACT:

A method, system, and computer-readable code for translating an input document into an Extensible Markup Language (XML) dialect which is well-formed, such that automated, dynamically-selected transformations (such as those that will indicate a user's current context) can be applied to the document. The new XML dialect indicates dynamically-selected document transformations that are desired. Further, a novel technique is provided for dynamically generating a Document Type Definition (DTD) to describe the new XML dialect, so that the XML document created in this dialect can subsequently be processed by an XML parser. In the preferred embodiment, the desired transformations account for a user's dynamic context, and this information is represented by the dynamically-generated XML dialect.

21 Claims, 6 Drawing figures

Previous Doc Next Doc Go to Doc#

A

L5: Entry 3 of 7 File: USPT Nov 5, 2002

US-PAT-NO: 6476833
DOCUMENT-IDENTIFIER: US 6476833 B1

TITLE: Method and apparatus for controlling browser functionality in the context of an application

DATE-ISSUED: November 5, 2002

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Moshfeghi; Mehran	Sunnyvale	CA		

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
Koninklijke Philips Electronics N.V.	Eindhoven			NL	03

APPL-NO: 09/ 281393 [PALM]
DATE FILED: March 30, 1999

INT-CL: [07] G06 F 3/74

US-CL-ISSUED: 345/854; 345/762
US-CL-CURRENT: 345/854; 345/762

FIELD-OF-SEARCH: 345/745, 345/742, 345/751, 345/759, 345/839, 345/744, 345/968, 345/853, 345/854, 345/762

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

Search Selected Search ALL Clear

	PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
☐	5784564	July 1998	Camaisa et al.	
☐	5796395	August 1998	De Hond	345/744
☐	6177936	January 2001	Cragun	345/781
☐	6266060	July 2001	Roth	345/853

FOREIGN PATENT DOCUMENTS

h e b b g e e e f c e e ge

	PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
☐	5530852	June 1996	Meske, Jr. et al.	709/206

OTHER PUBLICATIONS

SoftQuad:Products:Panorama Publishing Suite (visited Oct. 23, 1997) <http://www.softquad.com/products/panorama/>.
Computing Art Inc. (Visited Jul. 23, 1997) <http://mindlink.net/c-art-w/>.
Balise 3.0 Product Brief (03/96) <http://www.balise.com/current/pbrief2.htm>.
OmniMark LE Description (visited Oct. 6, 1997) <http://www.omnimark.com/release/omle/31/omlesamp.htm>.
Robin Cover. Public SGML Software (last modified Sep. 23, 1997) <http://www.sil.org/sgml/publicSW.html>.
Robin Cover. SGML: First Freeware SGML Viewer for the World Wide Web (last modified Jul. 11, 1997) <http://www.sil.org/sgml/free-pan.html>.
dtd2html (visited Oct. 6, 1997) <http://www.oac.uci.edu/indiv/ehood/perlSGML/doc/html/dtd2html.html>.
Grey Matter (visited Nov. 26, 1997) <http://www.greymatter.co.uk/gmWEB/items/00000370.htm>.
Multidoc Pro.RTM. Product Brief (visited Nov. 26, 1997) <http://gopher.sil.org/sgml/multidocPro.html>.
SoftQuad, Inc., SoftQuad HotMetal Pro 3.0 for Microsoft Windows, Second Edition (Jun. 1996), ISBN-1896172-40-7, Product No. SQGBO 04F6F 4C598.

ART-UNIT: 216

PRIMARY-EXAMINER: Hong; Stephen S.

ATTY-AGENT-FIRM: Oblon, Spivak, McClelland, Maier & Neustadt, P.C.

ABSTRACT:

A method, apparatus, and computer program product for providing a graphical user interface for creating and editing a mapping of structured information to different structured information, which allows a user to interactively define the mapping. The present invention operates as a user tool by accepting interactive input from a user of a source input, processing the input to display the source input in a format for accepting user commands to create or edit a transformation map of source components to target components. Interactive user input is accepted for selection of an input file to be transformed and selection of a transformation map for the requested transformation. Interactive user input is accepted for processing for selection of individual components of the first structured information format for mapping, and for selection of options for the target components. Exemplary options for the target components are a null value, the source component itself, a single selected target component, or plural selected target components. Interactive user input is accepted for processing to assign attribute values to components of the second structured information format. Exemplary options for the sources of attribute values are attribute values obtained from the source components, system attribute values, no value, attribute values input interactively by the user through the user interface, and content of element. Interactive user input is then accepted and processed to initiate processing of a transformation of the source input file in the first structured information format to a target output file in the second structured information format.

55 Claims, 48 Drawing figures

L5: Entry 6 of 7 File: USPT Jul 4, 2000

US-PAT-NO: 6085196
DOCUMENT-IDENTIFIER: US 6085196 A

TITLE: Object-oriented system and computer program product for mapping structured information to different structured information

DATE-ISSUED: July 4, 2000

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Motoyama; Tetsuro	Cupertino	CA		
Fong; Avery	Hayward	CA		
Bhatnagar; Anurag	Sunnyvale	CA		

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
Ricoh Company, Ltd.	Tokyo			JP	03
Ricoh Corporation	San Jose	CA			02

APPL-NO: 08/ 997482 [PALM]
DATE FILED: December 23, 1997

PARENT-CASE:
CROSS-REFERENCES TO RELATED APPLICATIONS This application is related to and being concurrently filed with two other patent applications: U.S. patent application Ser. No. 08/997,707, now U.S. Pat. No. 6,009,436, entitled "Method and Apparatus For Mapping Structured Information to Different Structured Information" and U.S. patent application Ser. No. 08/997,705, entitled "Method and Apparatus For Providing a Graphical User Interface For Creating and Editing a Mapping of a First Structural Description to a Second Structural Description", now pending, each filed on Dec. 23, 1997, and incorporated herein by reference .

INT-CL: [07] G06 F 17/30

US-CL-ISSUED: 707/102
US-CL-CURRENT: 707/102

FIELD-OF-SEARCH: 707/102

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

h e b b g e e e f c e e ge

	PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
☐	5530852	June 1996	Meske, Jr. et al.	709/206
☐	5832496	November 1998	Anand et al.	707/102
☐	5870746	February 1999	Knutson et al.	707/101
☐	5895476	April 1999	Orr et al.	707/517
☐	5907704	May 1999	Gudmundson et al.	395/701
☐	5909684	June 1999	Nelson	707/103
☐	5913065	June 1999	Faustini	395/703
☐	5928323	July 1999	Gosling et al.	709/203
☐	5944783	August 1999	Nieten	709/202
☐	5956737	September 1999	King et al.	707/517
☐	6012066	August 1999	Discount et al.	707/143

OTHER PUBLICATIONS

OmniMark LE Description (visited Oct. 6, 1997)
<http://www.omnimark.com/release/omle/31/omlesamp.htm>.
Robin Cover. Public SGML Software (last modified Sep. 23, 1997)
<http://www.sil.org/sgml/publicSW.html>.
Robin Cover. SGML: First Freeware SGML Viewer for the World Wide Web (last modified Jul. 11, 1997) <http://www.sil.org/sgml/free-pan.html>.
dtd2html (visited Oct. 6, 1997)
<http://www.oac.uci.edu/indiv/ehood/perISGML/doc/htmI/dtd2htmI.htmI>.
Grey Matter (visited Oct. 6, 1997)
<http://www.greymatter.co.uk/grmWEB/items/00000370.htm>.
Multidoc Pro.RTM. Product Brief (visited Nov. 26. 1997)
<http://gopher.sil.org/sgml/multidocPro.html>.
SoftQuad, Inc., SoftQuad HotMetal Pro 3.0 for Microsoft Windows, Second Edition (Jun. 1996), ISBN--1896172-40-7, Product No. SQGBO 04F6F 4C598.
SoftQuad:Products:Panorama Publishing Suite (visited Oct. 23, 1997)
<http://www.softquad.com/products/panorama/>.
Computing Art Inc. (Visited Jul. 23, 1997) <http://mindlink.net/c-art-w/>.
Balise 3.0 Product Brief (Mar. 1996) <http://www.balise.com/current/pbrief2.htm>.

ART-UNIT: 271

PRIMARY-EXAMINER: Amsbury; Wayne

ATTY-AGENT-FIRM: Oblon, Spivak, McClelland, Maier & Neustadt, P.C.

ABSTRACT:

An object-oriented system and computer program product for mapping structured information to different structured information, which allows a user to interactively define the mapping. The present invention operates as an object-oriented user tool by accepting interactive input from a user of a source input, by processing the input to display the source input in a format for accepting and processing user commands to create or edit a transformation map of source components to target components. Interactive user input is then accepted and processed for selection of an input file to be transformed and selection of a transformation map to be used for the requested transformation. Interactive user input is accepted and processed for selection of individual components of the first

structured information format for mapping, and for selection of options for the target components. Exemplary options for the target components are a null value, the source component itself, a single selected target component, or plural selected target components. Interactive user input is accepted for processing to assign attribute values to components of the second structured information format. Exemplary options for the sources of attribute values are attribute values obtained from the source components, system attribute values, no value, attribute values input interactively by the user, and content of element. Interactive user input is then accepted and processed to initiate processing of a transformation of the source input file in the first structured information format to a target output file in the second structured information format.

31 Claims, 47 Drawing figures

Previous Doc Next Doc Go to Doc#

First Hit Fwd Refs Previous Doc Next Doc Go to Doc#

End of Result Set

Generate Collection Print

L5: Entry 7 of 7 File: USPT Dec 28, 1999

US-PAT-NO: 6009436
DOCUMENT-IDENTIFIER: US 6009436 A

TITLE: Method and apparatus for mapping structured information to different structured information

DATE-ISSUED: December 28, 1999

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Motoyama; Tetsuro	Cupertino	CA		
Fong; Avery	Hayward	CA		
Bhatnagar; Anurag	Sunnyvale	CA		

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
Ricoh Company, Ltd.	Tokyo			JP	03
Ricoh Corporation	San Jose	CA			02

APPL-NO: 08/ 997707 [PALM]
DATE FILED: December 23, 1997

PARENT-CASE:
CROSS-REFERENCES TO RELATED APPLICATIONS This application is related to and being concurrently filed with two other patent applications: U.S. patent application Ser. No. 08/997,482, entitled "Object-Oriented System and Computer Program Product For Mapping Structured Information to Different Structured Information" and U.S. patent application Ser. No. 08/997,705, entitled "Method and Apparatus For Providing a Graphical User Interface For Creating and Editing a Mapping of a First Structural Description to a Second Structural Description", each filed on Dec. 23, 1997, and incorporated herein by reference.

INT-CL: [06] G06 F 17/30

US-CL-ISSUED: 707/102; 707/4, 707/100, 707/101, 706/59, 706/60
US-CL-CURRENT: 707/102; 706/59, 706/60, 707/100, 707/101, 707/4

FIELD-OF-SEARCH: 707/102, 707/2, 707/4, 707/100, 707/101, 706/59, 706/60

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

Search Selected Search ALL Clear

h e b b g e e e f c e e ge

	PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
☐	4930071	May 1990	Tou et al.	707/4
☐	5146406	September 1992	Jensen	704/9
☐	5193185	March 1993	Lanter	707/101
☐	5291602	March 1994	Baker et al.	707/524
☐	5487165	January 1996	Tsay et al.	707/102
☐	5717950	February 1998	Yamaguchi et al.	710/8
☐	5765154	June 1998	Horikiki et al.	707/10
☐	5799325	August 1998	Rivette et al.	707/500
☐	5819252	October 1998	Benson et al.	707/1
☐	5819265	June 1998	Ravin et al.	704/5
☐	5848386	December 1998	Motoyama	704/5
☐	5862325	January 1999	Reed et al.	395/200.31
☐	5878406	April 1999	Noyes	706/55
☐	5878419	March 1999	Carter	707/10
☐	5893066	April 1999	Hong	704/500
☐	5893109	April 1999	DeRose et al.	707/104
☐	5918240	June 1999	Kupiec et al.	707/531
☐	5918248	June 1999	Newell et al.	711/147

ART-UNIT: 277

PRIMARY-EXAMINER: Lintz; Paul R.

ASSISTANT-EXAMINER: Fleurantin; Jean Bolte

ATTY-AGENT-FIRM: Oblon, Spivak, McClelland, Maier & Neustadt, P.C.

ABSTRACT:

A method, apparatus, and computer program product for mapping a first structured information format to a second structured information format, which allows a user to interactively define the mapping. The present invention operates as a user tool by accepting interactive input from a user of a source input, by processing the input to display the source input in a format for accepting and processing user commands to create or edit a transformation map of source components to target components. Interactive user input is then accepted and processed for selection of an input file to be transformed and selection of a transformation map to be used for the requested transformation. Interactive user input is accepted and processed for selection of individual components of the first structured information format for mapping, and for selection of options for the target components. Exemplary options for the target components are a null value, the source component itself, a single selected target component, or plural selected target components. Interactive user input is accepted for processing to assign attribute values to components of the second structured information format. Exemplary options for the sources of

h e b b g e e e f c e e ge

attribute values are attribute values obtained from the source components, system attribute values, no value, attribute values input interactively by the user, and content of element. Interactive user input is then accepted and processed to initiate processing of a transformation of the source input file in the first structured information format to a target output file in the second structured information format.

49 Claims, 45 Drawing figures

Previous Doc Next Doc Go to Doc#

First Hit Fwd Refs Previous Doc Next Doc Go to Doc#

Generate Collection Print

L5: Entry 1 of 7 File: USPT Jul 13, 2004

US-PAT-NO: 6763499
DOCUMENT-IDENTIFIER: US 6763499 B1

TITLE: Methods and apparatus for parsing extensible markup language (XML) data streams

DATE-ISSUED: July 13, 2004

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Friedman; Greg S	Redmond	WA		
Lovett; Christopher J	Woodinville	WA		
Zeng; Nanshan	Redmond	WA		

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
Microsoft Corporation	Redmond	WA			02

APPL-NO: 09/ 361784 [PALM]
DATE FILED: July 26, 1999

INT-CL: [07] G06 F 17/22

US-CL-ISSUED: 715/513
US-CL-CURRENT: 715/513

FIELD-OF-SEARCH: 703/511, 717/114, 715/513, 715/501.1, 715/500.1, 715/531

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

Search Selected Search ALL Clear

	PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
☐	5416896	May 1995	Motoyama	707/514
☐	5483629	January 1996	Motoyama et al.	707/514
☐	6249844	June 2001	Schloss et al.	707/513
☐	6330569	December 2001	Baisley et al.	707/203
☐	6463440	October 2002	Hind et al.	707/102
☐	2003/0005410	January 2003	Harless	717/114

h e b b g e e e f c e e ge

OTHER PUBLICATIONS

The American Heritage Dictionary of the English Language, Third Edition, .COPYRGT. 1992 Houghton Mifflin Company. "Mapping".*
"SAX Implementation of a Namespace Parser Filter" by Cowan, http://ccil.org/.about.cowan/XML. Dec. 6, 1998.*
"Fundamentals of Computer Science I, Stacks and Queues," "http://www.math.grin.edu/courses/Scheme/spring-1998/stacks-and-queues. html".*
"Namespaces in XML".quadrature..quadrature.World Wide Web Consortium Jan. 14, 1999 http://www.w3.org/TR/1999/REC-xml-names-19990114.*
Huck et al., "Jedi: Extracting and Synthesizing Information from The Web", IEEE, pp. 32-41, Aug. 1998.*
Suzuki et al., "Managing The Software Design Documents with XML", ACM, pp. 127-136, Sep. 1998.*
Royappa, "Implementing Catalog Clearinghouses with XML and XSL", ACM, pp. 616-621, Mar. 1999.*
Jeuring et al., "Generic Programming for XML Tools" http://archive.cs.uu.nl/pub/RUU/CS/techreps/CS-2002/2002-023.pdf.*
Baker, "An XML Namespace Alternative" http://www.biglist.com/lists/xsl-list/archives/199901/msg00076.html.*
"What is an Event-Based Interface" http://www.megginson.com/SAX/event.html.*
"Quick Start for SAX Application Writers" http://www.megginson.com/SAX/quickstart.html.*
Aho, Compilers, Compilers: Principles, Techniques, and tools, Mar. 1998.*
W3C (World Wide Web Consortium) Dom Level 2 Working Draft at http://www.w3.org/TR/1999/WD-DOM-Level-2-19990719, published on Jul. 19, 1999.

ART-UNIT: 2178

PRIMARY-EXAMINER: Hong; Stephen S.

ASSISTANT-EXAMINER: Queler; Adam

ATTY-AGENT-FIRM: Lee & Hayes, PLLC

ABSTRACT:

Various features enable an XML data stream to be parsed without the need to build a hierarchical tree structure for the XML document. In the described embodiment, the concept of an element or namespace stack is utilized as a way of organizing parsing activities and maintaining a definable place within the structure of the XML document. Various structures work together with the element or namespace stack to facilitate piecewise parsing of the XML data stream. One structure is a namespace hierarchy that is a collection of namespace objects that each represent a namespace specification that is encountered in the XML data stream. Each object includes a namespace prefix and an associated namespace specification. This structure creates a hierarchical organization that is used for mapping a particular encountered namespace specification into a unique value that represents both the namespace specification and an element tag in which the namespace specification occurs. Another structure is a dictionary collection that contains one or more dictionaries. Each dictionary is specifically associated with a namespace specification that is encountered in the XML data stream. The dictionaries contain entries for one or more tag names and each name's associated unique token. The token is returned and placed on the element stack along with another special value that enables the proper state to be maintained during processing of the XML data stream. The stack also includes a text accumulation buffer that can hold any text that is contained within an element (between the element tags). When an XML element is encountered, the element stack is used to organize parsing activities as the parser makes its way through the XML data stream.

h e b b g e e e f c e e ge

L5: Entry 1 of 7 File: USPT Jul 13, 2004

DOCUMENT-IDENTIFIER: US 6763499 B1
TITLE: Methods and apparatus for parsing extensible markup language (XML) data streams

Abstract Text (1):
Various features enable an XML data stream to be parsed without the need to build a hierarchical tree structure for the XML document. In the described embodiment, the concept of an element or namespace stack is utilized as a way of organizing parsing activities and maintaining a definable place within the structure of the XML document. Various structures work together with the element or namespace stack to facilitate piecewise parsing of the XML data stream. One structure is a namespace hierarchy that is a collection of namespace objects that each represent a namespace specification that is encountered in the XML data stream. Each object includes a namespace prefix and an associated namespace specification. This structure creates a hierarchical organization that is used for mapping a particular encountered namespace specification into a unique value that represents both the namespace specification and an element tag in which the namespace specification occurs. Another structure is a dictionary collection that contains one or more dictionaries. Each dictionary is specifically associated with a namespace specification that is encountered in the XML data stream. The dictionaries contain entries for one or more tag names and each name's associated unique token. The token is returned and placed on the element stack along with another special value that enables the proper state to be maintained during processing of the XML data stream. The stack also includes a text accumulation buffer that can hold any text that is contained within an element (between the element tags). When an XML element is encountered, the element stack is used to organize parsing activities as the parser makes its way through the XML data stream.

Application Filing Date (1):
19990726

Brief Summary Text (4):
Extensible Markup Language (XML) is a meta-markup language that provides a format for describing structured data. XML is similar to HTML in that it is a tag-based language. By virtue of its tag-based nature, XML defines a strict tree structure or hierarchy. XML is a derivative of Standard Generalized Markup Language (SGML) that provides a uniform method for describing and exchanging structured data in an open, text-based format. XML utilizes the concepts of elements and namespaces. Compared to HTML, which is a display-oriented markup language, XML is a general-purpose language used to represent structured data without including information that describes how to format the data for display.

Brief Summary Text (9):
One exemplary usage of XML is the exchange of data between different entities, such as client and server computers, in the form of requests and responses. A client might generate a request for information or a request for a certain server action, and a server might generate a response to the client that contains the information or confirms whether the certain action has been performed. The contents of these requests and responses are "XML documents", which are sequences of characters that comply with the specification of XML. Part of the document exchange process between

clients and servers involves parsing the XML documents when they are received. In many cases, it is convenient to represent these XML documents in memory as a hierarchical tree structure. Once the hierarchical tree structure is built, the actual parsing process can begin. Consider the following exemplary XML code:

Brief Summary Text (12):
FIG. 1 shows how the structure of the above code can be represented in a hierarchical tree structure. In FIG. 1, all of the elements or nodes are set out in an exemplary tree that represents the XML document. Such a structure is typically constructed in memory, with each node containing all data necessary for the start and end tags of that node.

Brief Summary Text (15):
XML parsers typically read XML files or data streams and construct a hierarchically structured tree, such as the one appearing in FIG. 1, as a data structure in memory. The XML parser then typically hands off this data structure data to viewers and other applications for processing. So, in the example XML code discussed above, a parser would first build the entire tree structure that is shown in FIG. 1 prior to interpreting the contents of the document. Only after the entire tree structure was built in memory would the parser begin to interpret the document.

Brief Summary Text (16):
One problem that is associated with XML parsers such as this is that they have to build an entire hierarchically structured tree in memory before interpreting the contents of the document. This approach is not efficient because of the demands it places on the memory that is required to store the tree structure and the speed with which information can be conveyed to a client. For example, this type of approach is not efficient for an application that is doing work in connection with a large quantity of XML data that might be streaming in at a relatively slow speed. Consider, for example, that a client asks a server for a list of all messages of a certain type that are in a certain folder. The entire message list is going to be returned by the server as one large data stream. If the client has to wait for the entire message list to be returned from the server, then the client cannot begin to display any portion of the list until all of the data has been received. Furthermore, this process requires the parser to make at least two passes over the data; the first pass to build the tree structure, and the second pass to traverse the nodes of the tree to interpret the contents of the document. This approach requires a large memory overhead (for storing the XML data and building the hierarchical tree structure) which, in turn, impacts the speed with which responses can be used by client applications.

Brief Summary Text (19):
Various features of the invention provide methods and systems for parsing an XML data stream that do not require that an entire hierarchical tree structure be built and stored in memory in conjunction with parsing activities.

Brief Summary Text (21):
A namespace hierarchy is defined as the XML data stream is received. In one described embodiment, the namespace hierarchy is a collection of namespace objects that each represent a namespace specification that is encountered in the XML data stream. Each object includes a namespace specification and, in the case of non-default namespace, a namespace prefix. Within the hierarchy, namespace objects refer to their parent. The namespace hierarchy is used for mapping a particular encountered namespace specification into a unique value that represents both the namespace specification and an element tag in which the namespace occurs.

Drawing Description Text (2):
FIG. 1 is an exemplary hierarchical tree structure that represents an XML document that is discussed in the "Background" section.

Drawing Description Text (7):
FIG. 6 is a diagram of a namespace hierarchy in accordance with one embodiment of the invention.

Drawing Description Text (13):
FIG. 12 is a diagram of a hypothetical hierarchical tree that represents an XML data stream.

Drawing Description Text (14):
FIG. 13 is a diagram of a namespace hierarchy that corresponds to the FIG. 12 hierarchical tree.

Drawing Description Text (15):
FIG. 14 is a diagram of an element stack that corresponds to the namespace hierarchy of FIG. 13.

Detailed Description Text (3):
A system as described below enables an XML data stream to be parsed without the need to build a hierarchical tree structure for an XML document. The XML data stream is parsed as the data stream is received. This saves memory overhead and increases the speed with which data can be provided to the client or application. In the described embodiment, an element or namespace stack is utilized as a way of organizing parsing activities and maintaining a definable place within the structure of the XML document. The element or namespace stack has a plurality of frames that are used to hold data during the parsing activities.

Detailed Description Text (4):
Various other structures work together with the element or namespace stack to facilitate piecewise parsing of the XML data stream as it is received. For example, a namespace hierarchy is built and maintained as the XML data stream is received. The namespace hierarchy is a collection of namespace objects that each represent a namespace that is encountered in the XML data stream. Each object includes a namespace prefix and an associated namespace specification or value. Some of the objects include a reference to another namespace object that is its parent. This structure creates a hierarchical organization that is used for mapping a particular encountered namespace specification into a unique value that represents both the namespace specification and an element tag in which the namespace specification occurs.

Detailed Description Text (19):
FIG. 4 shows a high level flow diagram that generally describes the processing that takes place as XML data is received and parsed. The client receives a notification that is generated by the parser 18 that a new element has been encountered at step 100. This happens when a start tag is encountered. When notification is received, the new element is pushed onto the element stack (step 102). The element stack includes a plurality of frames, each of which includes a frame portion for holding a value that is uniquely associated with a namespace specification and an element tag, a frame portion for maintaining a special value that is associated with a defined namespace specification (discussed below), and a text buffer for accumulating any text that might be contained between an element's tags. At step 104, notification is received that a close tag has been encountered. Consequently, the top frame of the element stack is removed at 106. In connection with removing the top frame, any text that has been accumulated in the text buffer can be processed. When the text is processed, it is provided to the application 12. Hence, data can be provided to the application before the entire XML data stream has been processed by the parser. In addition, the parser does not need to build a hierarchical tree structure in order to carry out parsing activities. This not only saves processing overhead, but reduces the time that an application must wait in order to receive and operate upon its data.

Detailed Description Text (21):
Various parsing support structures are utilized to facilitate the piece-wise parsing described above. These structures represent improvements over those structures that are used in connection with XML parsing activities that require a hierarchical tree structure that represents an XML document to be built and stored in memory before parsing activities begin. Exemplary parsing support structures include an element stack, a namespace hierarchy, and a dictionary collection.

Detailed Description Text (24):
Namespace Hierarchy

Detailed Description Text (25):
FIG. 6 shows a namespace hierarchy 300 that is built during parsing of the XML data stream. The namespace hierarchy is used to organize namespace specifications so that they can be mapped into a unique token value that is placed on the element stack during parsing. The illustrated namespace hierarchy has two exemplary namespace objects 302, 304. Namespace object 302 is the parent of namespace object 304. Namespace object 302 has no parent namespace object. Each namespace object contains a reference to a dictionary that is associated with the namespace object. Dictionaries are discussed below in more detail.

Detailed Description Text (26):
The process of building a namespace hierarchy having namespace objects involves looking at the XML data stream as it is received and extracting certain information from it. As a simple example, consider the following excerpt of an XML data stream:

Detailed Description Text (28):
FIG. 7 shows a flow diagram that describes steps in a method to build namespace hierarchy 300 in FIG. 6. When the above XML data stream is received, step 400 determines whether there are any attribute declarations in the multistatus element. If there are no attribute declarations, then the method branches to a method (step 420) that retrieves a unique token for the namespace prefix of the element ("D" in this case) and the element name ("multistatus"). If there are attribute declarations, as there are here, step 402 gets the first attribute declaration (i.e. xmlns:D="DAV") and step 404 determines whether it is a namespace declaration or specification. If it is not a namespace declaration or specification, the method branches to step 406 which determines whether there are any more attribute declarations. If there are, steps 402 and 404 are repeated. If not, then step 406 branches to the method (step 420) that defines a unique token for the namespace prefix of the element.

Detailed Description Text (30):
The namespace hierarchy is a useful structure to keep track of the namespace specifications that might be declared in the XML data stream. As namespace specifications can be declared inside any element within the XML tree, keeping track of their hierarchical occurrence and place in the hierarchy becomes important. That is, maintaining the state of the namespace hierarchy during parsing activities is important if proper parsing is to take place.

Detailed Description Text (36):
There, step 600 encounters an element tag. During parsing activities there is state data that gets maintained across the parsing process. One piece of the state data that gets maintained is defined as the "top namespace." The "top namespace" is a variable that is used to track namespace specifications within the namespace hierarchy. Initially, there is no "top namespace" because parsing activities have not yet begun. When an element tag is encountered, step 602 stores the current "top namespace" in a local variable called "oldTopNamespace". The reason that it is stored locally is because the processing of an element will change the value of the "top namespace". By storing it locally, reference can later be made to the current

"top namespace" as it existed before the element was processed. Step 604 determines whether there are any attribute declarations in the element (this step is identical to step 400 in FIG. 7). If there are attribute declarations in the element, then processing takes place as was described in connection with FIG. 7. For namespace declarations this results in building the namespace hierarchy. If there are no attribute declarations or, if all of the attribute declarations have been processed as set forth in FIG. 7, then the method creates a unique token for the current element (element name) and its prefix.

Detailed Description Text (38):
Recall that in the example given above, a namespace hierarchy 300 (FIG. 6) and a dictionary collection 500 (FIG. 8) were created for the excerpted XML data stream below:

Detailed Description Text (40):
FIG. 10 shows a flow diagram that describes an exemplary method of converting the namespace prefix and element name into a unique token. Step 700 traverses the namespace objects to find an expanded namespace specification that has a prefix that matches the prefix extracted in step 606 (FIG. 9). In this example, step 700 looks through the individual namespace objects in namespace hierarchy 300 (FIG. 6) to find the prefix "D" that matches the "D" extracted from the current element tag. The first namespace object that is checked is namespace object 304. This object contains a prefix "G" which does not match the "D" from the current element tag. Step 700 then moves onto the parent namespace object 302. Here there is a match between the prefixes "D". Step 702 then uses the expanded namespace specification for the prefix that matched (i.e. "DAV") and references a data map to find a dictionary for that expanded namespace specification. Here, step 702 gets the dictionary from the dictionary reference that the namespace object contains. Step 704 then searches the dictionary to locate the tag name and its associated token. In this example, step 704 looks for the "multistatus" entry in the dictionary. When it finds this entry at step 706, it returns the token (step 710) that is associated with the "multistatus" entry--here "DAV-MULTISTATUS". If the tag name is not found, step 708 returns an "Unknown" message. The "DAV-MULTISTATUS" token is a constant and is guaranteed to be unique no matter how many namespace specifications appear in the XML data stream.

Detailed Description Text (42):
Returning to FIG. 9, after converting the namespace prefix and element name into a unique token, step 610 pushes the element onto the element stack (FIG. 5). The element stack has individual frames 202, 204, and 206. Each frame has a plurality of frame portions. Frame portions are shown for frame 202 at 202a, 202b, and 202c. Frame portion 202a holds the element tag token that is returned from step 710 (FIG. 10). Along with the element tag token, frame portion 202b holds a variable called "oldTopNamespace". Recall that the "oldTopNamespace" variable is a local variable that holds the value of the current "top" namespace (step 602 in FIG. 9). The current "top" namespace value can, however, change because each time a namespace specification is encountered in the XML data stream, the namespace hierarchy grows by a corresponding namespace object. Each time a namespace object is added, it becomes the new current "top" namespace (step 418 in FIG. 7). The "oldTopNamespace" variable is used to maintain the state of the namespace hierarchy as the parser moves into and out of different layers of the XML data stream. For example, when a close tag is encountered for an element, any namespace specifications that were declared in that element are no longer valid. This is because the namespaces have a life span that is only equal to the lifespan of the stack frame that represents the element in which they are declared. Accordingly, as will become apparent below, the "oldTopNamespace" variable provides a way to return the namespace hierarchy to the state that it was in before the top stack frame was pushed onto the stack.

Detailed Description Text (45):
FIG. 11 shows a flow diagram that describes steps in a method for close tag

processing. Step 800 encounters the close tag. When a close tag is encountered, the parser sends a notification to that effect to the client. When the notification is received, any text that is in the text accumulation buffer of the top frame of the element stack 200 (FIG. 5) is processed. This can involve displaying text for a client to view, or any other action that is contextually appropriate. Step 804 then determines whether the current "top" namespace is equal to the namespace in the top frame of the element stack. If they are not equal, then step 806 requests the parent of the current "top" namespace. The parent is stored in a variable "localparent" at step 808. Step 810 deletes the current "top" namespace and step 812 then sets the current top namespace to "localParent". Step 812 then returns to step 804 to compare the current "top" namespace with the namespace in the top frame of the element stack. If they are equal, then step 814 removes the top frame from the element stack and step 816 looks for the next element tag. This process keeps rolling back through the namespace hierarchy so that it can be placed into the state that it was in when that element was pushed onto the element stack.

Detailed Description Text (47):
As a simple example that illustrates how the element stack is used during parsing activities, and how the state of the namespace hierarchy is maintained through the use of the element stack, consider FIGS. 12-14. FIG. 12 is a diagram that represents a hypothetical hierarchical tree that represents an XML data stream. Each element is represented by a letter--here letters A through C. Namespace declarations or specifications take place at elements A (ns1 and ns2) and B (ns3).

Detailed Description Text (48):
During processing of the XML data stream, the element tag for A is encountered. When this happens, processing takes place as described in connection with FIG. 9. The current "top" namespace is stored in a local variable "oldTopNamespace" (step 602 in FIG. 9). Here, since there is no current "top" namespace, nothing is stored. In addition, the element is checked for namespace declarations and if there are any, namespace objects are built (FIG. 7) and arranged into a namespace hierarchy. Thus here, there are two namespaces declared--"ns1" and "ns2" in element A. After processing through the namespace specifications, namespace objects for "ns1" and "ns2" are created as shown in FIG. 13. The namespace prefix and element name are converted into a unique token ("A.sub.token ") and the element is pushed onto the element stack. At this point in FIG. 14, only the lower frame holding "A.sub.token " would be on the element stack. After processing through element A, the following values exist for "oldTopNamespace" and current "top" namespace:

Detailed Description Text (49):
When element B is encountered, processing takes place as described above. First, the current "top" namespace specification "ns2" is stored in "oldTopNamespace". Since element B includes a new namespace declaration (i.e. "ns3"), it is processed into the namespace hierarchy of FIG. 13 and "ns3" becomes the current "top" namespace. Element B's namespace prefix and element name are converted into a unique token (step 608) and the element is pushed onto the element stack (step 610). At this point in the processing, the following values exist for "oldTopNamespace" and current "top" namespace:

Detailed Description Paragraph Table (7):
DWORD dwSize The size of this structure in bytes. DWORD dwType The node type. DWORD dwSubType The node sub type. BOOL fTerminal True if this node cannot have any children and so BeginChildren and EndChildren are guaranteed not to be called for this node. const WCHAR* pwcText This is either a tag name, or a PCDATA text value. The lifetime of this pointer is the lifetime of the CreateNode call. Note that Element/Attribute/PI tag names or certain attribute values (of type ID, NMTOKEN, ENTITY, or NOTATION), may have namespace prefixes. ULONG ulLen The length of the element or attribute name. ULONG ulNsPrefixLen The length of the namespace prefix, when present. PVOID pNode This field can be used by the NodeFactory to RETURN an object representing the node. Since this is PVOID, the NodeFactory can build ANY

tree representation it wants - even a raw struct hierarchy. PVOID * pReserved For private communication between factories.

CLAIMS:

11. The method of claim 10 further comprising building a namespace hierarchy of namespace specifications that are declared in the XML data stream, individual namespace specifications in the hierarchy pointing to other namespace specifications in the hierarchy; and wherein said comparing comprises searching through the namespace hierarchy.

14. The computer-readable medium of claim 13 further comprising building a namespace hierarchy of namespace specifications that are declared in the XML data stream, individual namespace specifications in the hierarchy pointing to other namespace specifications in the hierarchy; and wherein said comparing comprises searching through the namespace hierarchy.

19. A parsing support structure embodied on a computer-readable media, for parsing an XML data stream comprising: an element stack; a plurality of frames within the element stack; and each frame within the element stack comprising: a first frame portion that holds an element tag token, each element tag token uniquely representing an element tag and a namespace specification that is declared within the element tag; and a second frame portion that holds a variable associated with a current top namespace, wherein the current top namespace changes as namespace specifications are encountered in the XML data stream and the second frame portion is used to maintain state of a namespace hierarchy as an XML parser moves into and out of different layers of the XML data stream, wherein said parsing support structure permits parsing of the XML data stream to begin without requiring an XML tree structure, comprising an XML document embodied by the XML data stream, to be built.

21. The parsing support structure of claim 19 further comprising: a namespace hierarchy defined by one or more namespace objects; each namespace object having a prefix, a dictionary reference, and an expanded namespace specification associated therewith; and wherein said dictionary reference references a dictionary which, for a given namespace specification, contains said element tag tokens.

22. A parsing support structure embodied on a computer-readable media, for parsing an XML data stream comprising: a namespace hierarchy comprising a plurality of namespace objects, each namespace object having a prefix and an expanded namespace specification associated therewith; and a dictionary collection comprising a dictionary for each namespace object in the namespace hierarchy, wherein each dictionary comprises a tag name field for holding tag names that can be encountered in the XML data stream and a token field for holding token values for each associated tag name appearing in the tag name field, the token values uniquely representing an expanded namespace specification and an associated tag name, wherein the namespace hierarchy and dictionary collection are used in a stack-based XML parsing process in which parsing of the XML data stream begins without requiring an XML tree structure, comprising an XML document embodied by the XML data stream, to be built.

24. A method of parsing an XML data stream comprising: receiving an XML data stream containing a namespace prefix and an associated element tag name, the element tag name being associated with an element tag; converting the namespace prefix and the element tag name into a token that uniquely represents a namespace specification that is associated with the namespace prefix and the element tag; defining a stack that is configured to receive one or more tokens during parsing of the XML data stream; and placing a token on the stack, wherein the stack comprises a plurality of frames each of which includes a token field for holding a token value, a namespace field for holding a namespace value, and a text field for holding any

text that is associated with a particular element tag, and further comprising: for each element tag that is encountered in the XML data stream: storing a value for a current top namespace, if any, in a defined variable; and after said storing, determining whether the element tag contains any namespace declarations and, if so, building a namespace hierarchy comprising namespace objects for each declared namespace, the most recently created namespace object defining the current top namespace; after said placing, determining whether a next element tag is a close element tag and if so: processing any text that is contained in the text field of the top stack frame; determining whether the value that was stored in the defined variable for the current top namespace matches the namespace value that is held in the namespace field for the stack frame and if there is match, removing the top stack frame from the stack; and if there is not a match, defining a new value for the defined variable and determining whether the new value matches the namespace value that is held in the namespace field for the stack frame.

Previous Doc Next Doc Go to Doc#

L6: Entry 1 of 6 File: USPT Jul 13, 2004

DOCUMENT-IDENTIFIER: US 6763499 B1
TITLE: Methods and apparatus for parsing extensible markup language (XML) data streams

Application Filing Date (1):
19990726

Detailed Description Text (19):
FIG. 4 shows a high level flow diagram that generally describes the processing that takes place as XML data is received and parsed. The client receives a notification that is generated by the parser 18 that a new element has been encountered at step 100. This happens when a start tag is encountered. When notification is received, the new element is pushed onto the element stack (step 102). The element stack includes a plurality of frames, each of which includes a frame portion for holding a value that is uniquely associated with a namespace specification and an element tag, a frame portion for maintaining a special value that is associated with a defined namespace specification (discussed below), and a text buffer for accumulating any text that might be contained between an element's tags. At step 104, notification is received that a close tag has been encountered. Consequently, the top frame of the element stack is removed at 106. In connection with removing the top frame, any text that has been accumulated in the text buffer can be processed. When the text is processed, it is provided to the application 12. Hence, data can be provided to the application before the entire XML data stream has been processed by the parser. In addition, the parser does not need to build a hierarchical tree structure in order to carry out parsing activities. This not only saves processing overhead, but reduces the time that an application must wait in order to receive and operate upon its data.

Detailed Description Text (59):
With the xmlns declaration attributes having been processed and the namespace stack having been built, namespace scoping rules can be applied efficiently during subsequent parsing. Here, step 912 builds name objects that are used during parsing to maintain the current state of the namespace specifications. Consider the processing that takes place when the "item" element appearing just below is parsed:
##EQU1##

L5: Entry 5 of 7 File: USPT Aug 21, 2001

US-PAT-NO: 6279015
DOCUMENT-IDENTIFIER: US 6279015 B1

TITLE: Method and apparatus for providing a graphical user interface for creating and editing a mapping of a first structural description to a second structural description

DATE-ISSUED: August 21, 2001

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Fong; Avery	Hayward	CA		
Motoyama; Tetsuro	Cupertino	CA		
Bhatnagar; Anurag	Sunnyvale	CA		

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
Ricoh Company, Ltd.	Tokyo			JP	03
Ricoh Corporation	San Jose	CA			02

APPL-NO: 08/ 997705 [PALM]
DATE FILED: December 23, 1997

PARENT-CASE:
CROSS-REFERENCES TO RELATED APPLICATIONS This application is related to and being concurrently filed with two other patent applications: U.S. patent application Ser. No. 08/997,707, now U.S. Pat. No. 6,009,436 entitled "Method and Apparatus For Mapping Structured Information to Different Structured Information" and U.S. patent application Ser. No. 08/997,482, entitled "Object-Oriented System and Computer Program Product For Mapping Structured Information to Different Structured Information", each filed on Dec. 23, 1997, and incorporated herein by reference.

INT-CL: [07] G06 F 17/30

US-CL-ISSUED: 707/523; 707/513, 707/514
US-CL-CURRENT: 715/523; 715/513, 715/514

FIELD-OF-SEARCH: 707/101, 707/102, 707/103, 707/513, 707/514, 707/523, 707/540, 345/326, 345/339, 345/356

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

h e b b g e e e f c e e ge

L8: Entry 1 of 4 File: TDBD Sep 1, 2001

TDB-ACC-NO: NNRD44981

DISCLOSURE TITLE: Ordered and Optimized DOM Initialization

PUBLICATION-DATA:
IBM technical Disclosure Bulletin, September 2001, UK

ISSUE NUMBER: 449
PAGE NUMBER: 1535

PUBLICATION-DATE: September 1, 2001 (20010901)

CROSS REFERENCE: 0374-4353-0-449-1535

DISCLOSURE TEXT:

This invention describes a process and method to initialize Document Object Model (DOM) nodes in an orderly manner, when parsing an XML document that produces a DOM. This is done by adding a descriptor attribute to each XML tag to be used to indicate the order of initialization of the DOM nodes. Applications that use XML typically use a SAX parser to parse the XML into a DOM tree. A SAX parser is very powerful in that it allows an application to create a custom DOM of specific application classes. For each XML tag in the parsed XML document a Java object is created. These objects are linked together to form the DOM tree. In the current art, when the document parse has completed successfully, an application typically walks the DOM tree calling methods on each DOM node object to do further initialization. Tree walking to initialize a DOM is not always optimal. If the XML document is very large (thousands of nodes or larger), and very few nodes require initialization, it will be very expensive to walk the entire DOM tree and call an initialization method on a few nodes. Also, in some cases order of initialization is important, e.g. one node may need to be initialized before another. This invention introduces the notion of adding an initialization attribute to the XML grammar for each tag. The value of the attribute for a given node determines whether or not the node requires an initialization call. The value of "NONE", the default value, denotes the node does not require any initialization. For Example: This tag contains the value "CALL-INIT1" for the INIT attribute. This tag requires a special initialization. At the beginning of the parse process an empty hash table is created. The hash table is keyed by values of the "INIT" attribute for a node, we'll call this "INIT" value the ivalue. The hash table also keeps a reference to a stack for each ivalue. As the parser process encounters nodes with an "INIT" attribute, the ivalue is used as a key into the hash table. Using the ivalue key into the hash table a reference to the associated init stack is retrieved. If no stack has been c reated for a given ivalue, a new stack is created and the reference is stored in the hash table. For each node that contains an ivalue, a reference to the node is pushed on the associated stack in the hash table. When parsing is completed the hash table contains a list of stacks, and each stack contains one or more DOM nodes that require initialization. The initialization process then enumerates the hash table entries and retrieves the stack reference for each ivalue. Once the stack reference is obtained, the stack is processed in a loop that pops each DOM node reference off the stack nd calls the appropriate initialization method on the DOM node. This continues until the stack is empty.

When a node stack has completed processing, the process moves to the next hash table entry, until all hash table entries have been processed. The process described so far does not provide any particular order of initialization for DOM nodes. If an ordered initializaton of sets of nodes is needed, the contents of the hash table could be sorted into a vector in a way that based on sort order of the ivalue, sets of nodes are initialized in order.

Previous Doc Next Doc Go to Doc#

h e b b g e e e f c e e ge

A

L8: Entry 2 of 4 File: DWPI Jul 11, 2002

DERWENT-ACC-NO: 2002-635852
DERWENT-WEEK: 200313

TITLE: Goods purchase order management method in internet, involves comparing existing contract terms and terms retrieved from order tag values and placing orders in case of equality

INVENTOR: MIKULINSKY, O; YEHIA, R ; YIN, J

PATENT-ASSIGNEE: PARTNER COMMUNITY INC (PARTN)

PRIORITY-DATA: 2001US-0757227 (January 9, 2001)

PATENT-FAMILY:

PUB-NO	PUB-DATE	LANGUAGE	PAGES	MAIN-IPC
US 20020091579 A1	July 11, 2002		026	G06F017/60

APPLICATION-DATA:

PUB-NO	APPL-DATE	APPL-NO	DESCRIPTOR
US20020091579A1	January 9, 2001	2001US-0757227	

INT-CL (IPC): G06 F 17/60

RELATED-ACC-NO: 2002-635848;2003-138837

ABSTRACTED-PUB-NO: US20020091579A
BASIC-ABSTRACT:

NOVELTY - An order received from a consumer partner is parsed into tag values representing order governing terms on existence of contract between consumer and provider partners (102). Retrieved terms are compared with contract terms. The orders are either placed to the providers or a notification is sent to the partners in case of equality or inequality between the terms, respectively.

DETAILED DESCRIPTION - INDEPENDENT CLAIMS are included for the following:

(1) Business order management method;

(2) Computer readable medium storing order management program; and

(3) Centralized processing hub.

USE - For managing and correlating orders related to goods such as gas, oil, sewing

h e b b g e e e f c e e ge

machine, services such as ticketing, tolls, freight charges, shipping charges, power supply, water supply and contracts in internet, intranet, satellite, cellular network, received through PC, mainframe computer, PDA, cellular phone.

ADVANTAGE - Enables reliable and automatic negotiation of orders and contracts between partners.

DESCRIPTION OF DRAWING(S) - The figure shows the block diagram of the order management system illustrating data flow between partners.

Consumer and provider partners 102

ABSTRACTED-PUB-NO: US20020091579A
EQUIVALENT-ABSTRACTS:

CHOSEN-DRAWING: Dwg.2/11

DERWENT-CLASS: T01
EPI-CODES: T01-J11A; T01-N01A2F;

Previous Doc Next Doc Go to Doc#

FOREIGN-PAT-NO	PUBN-DATE	COUNTRY	US-CL
WO9715008	April 1997	WO	
WO9740446	October 1997	WO	
WO9832076	July 1998	WO	
WO9837480	August 1998	WO	
WO 9845793	October 1998	WO	
WO9848546	October 1998	WO	

OTHER PUBLICATIONS

"Look Ahead Filtering on Internet Content" IBM Technical Disclosure Bulletin, US IBM Corp. New York, vol. 40, No. 12, Dec. 1, 1997, pp. 143.

ART-UNIT: 2173

PRIMARY-EXAMINER: Nguyen; Cao H.

ATTY-AGENT-FIRM: Vodopia; John

ABSTRACT:

This invention includes methods and apparatus for browsing markup language documents from within the context of a client-server application running on an end-user device. Browser functionality, which is configured according to user profile information specifying each user's authorization and preferences, is embedded in the application, and can be activated by application controls. While some users have unrestricted authorization and access, others are restricted to certain browser functions and to certain allowed network resources. This restriction is enforced by preventing the browser functionality from generating network addresses that are not on a list of allowed network addresses also present in the user profile information. Network access restriction is achieved, in part, by filtering markup language documents before display to delete linking information that is not allowed. Document filtering methods are presented for Hypertext Markup Language (HTML) and extensible Markup Language (XML) documents. The document filtering methods are extendable to additional markup languages.

28 Claims, 8 Drawing figures

Previous Doc Next Doc Go to Doc#

h e b b g e e e f c e e ge

- US Pat. 6,615,266

- US Pat. 6,565,609

Ouahid et al., "Converting Web Pages into Well-formed XML Documents", Jun. 1999, 1999 IEEE International Conference on Communications, 1999,

vol. 1, pp. 676-680.

TDB-ACC-NO: NNRD440132: Generic XML Generator from Java Hash table,

IBM technical Disclosure Bulletin, December 2000, UK; ISSUE NUMBER: 440

PAGE NUMBER: 2180, PUBLICATION-DATE: December 1, 2000

CROSS REFERENCE: 0374-4353-0-440-2180.

United States Patent **5,794,053**
Doris, Jr., et al. **August 11, 1998**

Method and system for dynamic interface contract creation

Abstract

A system for creating user defined software interface contracts for sending host system data to any external system. The external systems can reside on any type of hardware platform. The invocation of a contract is controlled by a set of user defined conditions known as an event. The creation of event and contracts is completely under the control of the users of the external systems with little or no support of the host system developers to establish new interfaces to the external systems.

Inventors: **Doris, Jr.; Daniel Joseph** (Lebanon, NJ); **Solar; Donald Joseph** (Livingston, NJ)

Assignee: **Bell Communications Research, Inc.** (Morristown, NJ)

Appl. No.: **681234**

Filed: **July 22, 1996**

Current U.S. Class: **717/162**; 709/200

Intern'l Class: G06F 013/00; G06F 015/163

Results of Search in 1976 to present db for:
((SPEC/pars? AND SPEC/"tag value") AND SPEC/XML): 7 patents.
Hits 1 through 7 out of 7

Jump To

Refine Search | SPEC/pars? AND SPEC/"tag value" AND SPEC/XM

	PAT. NO.	Title
1	6,721,747	Method and apparatus for an information server
2	6,671,869	Method and apparatus for graphically programming a programmable circuit
3	6,643,652	Method and apparatus for managing data exchange among systems in a network
4	6,635,089	Method for producing composite XML document object model trees using dynamic data retrievals
5	6,438,540	Automatic query and transformative process
6	6,263,332	System and method for query processing of structured documents
7	6,182,029	System and method for language extraction and encoding utilizing the parsing of text data in accordance with domain parameters

6,182,029

Friedman **January 30, 2001**

System and method for language extraction and encoding utilizing the parsing of text data in accordance with domain parameters

Abstract

A computerized method for extracting information from natural-language text data includes parsing the text data to determine the grammatical structure of the text data and regularizing the parsed text data to form structured word terms. The parsing step, which can be performed in one or more parsing modes, includes the step of referring to a domain parameter having a value indicative of a domain from which the text data originated, wherein the domain parameter corresponds to one or more rules of grammar within a knowledge base related to the domain to be applied for parsing the text data. Preferably, the structured output is mapped back to the words in the original sentences of the text data input using XML tags.

Inventors: **Friedman; Carol** (Larchmont, NY)

Assignee: **The Trustees of Columbia University in the City of New York** (New York,

NY)

Appl. No.: **370329**

Filed: **August 6, 1999**

Current U.S. Class: **704/9**; 715/513; 715/531

Intern'l Class: G06F 017/27

6,513,059

Gupta , et al. **January 28, 2003**

Adaptive collaborative intelligent network system

Abstract

System and method for facilitating exchange of information on a computer network, such as the Internet. The system provides one or more context trees, with each tree including two or more connected nodes, each node being associated with one or more selected node objects. Associated with each node is a blackboard for receiving and making available for reading, messages concerning the node object, a knowledge base containing information, facts, constraints and-or rules (Rules) concerning the node object, and an inference engine providing at least one logical rule that can be used to infer a logical conclusion based on at least one Rule in the knowledge base. A tree has a collection of at least two mobile intelligent agents that are configured to facilitate exchange of information on a node object, between two agents or between a node and an agent. An agent may migrate from a first node to a second node connected to the first node. An agent, by subscribing to the Rules of a node, is permitted to post a message on, and to read a message posted on, a blackboard for the subscribed node. The collection of agents has at least one tree agent that has knowledge of nodes that are directlz connected to each node in the tree.

Inventors: **Gupta; Pradeep** (Milpitas, CA); **Kondratiev; Dmitri** (Moscow, RU)

Assignee: **Cambira Corporation** (Santa Clara, CA)

Appl. No.: **648299**

Filed: **August 24, 2000**

Current U.S. Class: **709/202**

Intern'l Class: G06F 015/16

Field of Search: 709/202,203,249

First Hit Fwd Refs Previous Doc Next Doc Go to Doc#

Generate Collection Print

L1: Entry 1 of 2 File: USPT Jan 28, 2003

US-PAT-NO: 6513059
DOCUMENT-IDENTIFIER: US 6513059 B1

TITLE: Adaptive collaborative intelligent network system

DATE-ISSUED: January 28, 2003

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Gupta; Pradeep	Milpitas	CA		
Kondratiev; Dmitri	Moscow			RU

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
Cambira Corporation	Santa Clara	CA			02

APPL-NO: 09/ 648299 [PALM]
DATE FILED: August 24, 2000

INT-CL: [07] G06 F 15/16

US-CL-ISSUED: 709/202
US-CL-CURRENT: 709/202

FIELD-OF-SEARCH: 709/202, 709/203, 709/249

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

Search Selected Search ALL Clear

	PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
☐	6134548	October 2000	Gottsman et al.	707/5
☐	6199099	March 2001	Gershman et al.	709/203
☐	6275859	August 2001	Weseley et al.	709/229
☐	6356905	March 2002	Gershman	707/10
☐	6401085	June 2002	Gershman et al.	707/4

ART-UNIT: 2155

h e b b g e e e f c e e ge

PRIMARY-EXAMINER: Eng; David Y.

ATTY-AGENT-FIRM: Schipper; John F.

ABSTRACT:

System and method for facilitating exchange of information on a computer network, such as the Internet. The system provides one or more context trees, with each tree including two or more connected nodes, each node being associated with one or more selected node objects. Associated with each node is a blackboard for receiving and making available for reading, messages concerning the node object, a knowledge base containing information, facts, constraints and-or rules (Rules) concerning the node object, and an inference engine providing at least one logical rule that can be used to infer a logical conclusion based on at least one Rule in the knowledge base. A tree has a collection of at least two mobile intelligent agents that are configured to facilitate exchange of information on a node object, between two agents or between a node and an agent. An agent may migrate from a first node to a second node connected to the first node. An agent, by subscribing to the Rules of a node, is permitted to post a message on, and to read a message posted on, a blackboard for the subscribed node. The collection of agents has at least one tree agent that has knowledge of nodes that are directlz connected to each node in the tree.

34 Claims, 8 Drawing figures

Previous Doc Next Doc Go to Doc#

First Hit Fwd Refs Previous Doc Next Doc Go to Doc#
End of Result Set

Generate Collection Print

L1: Entry 2 of 2 File: USPT Jan 30, 2001

US-PAT-NO: 6182029
DOCUMENT-IDENTIFIER: US 6182029 B1

TITLE: System and method for language extraction and encoding utilizing the parsing of text data in accordance with domain parameters

DATE-ISSUED: January 30, 2001

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Friedman; Carol	Larchmont	NY		

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
The Trustees of Columbia University in the City of New York	New York	NY			02

APPL-NO: 09/ 370329 [PALM]
DATE FILED: August 6, 1999

PARENT-CASE:
SPECIFICATION This application is a continuation-in-part of U.S. application Ser. No. 08/738,889 filed Oct. 28, 1996, now U.S. Pat. No. 6,055,494 which is incorporated herein by reference.

INT-CL: [07] G06 F 17/27

US-CL-ISSUED: 704/9; 707/513, 707/531
US-CL-CURRENT: 704/9; 715/513, 715/531

FIELD-OF-SEARCH: 704/9, 704/1, 704/10, 705/2, 705/3, 707/104, 707/513, 707/531, 707/532, 707/533, 707/534, 706/45, 706/46, 706/55, 706/59

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

Search Selected Search ALL Clear

	PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
☐	5799268	August 1998	Boguraev	704/9
☐	5832496	November 1998	Anand et al.	707/102
	6038668	March 2000	Chipman et al.	713/201

h e b b g e e e f c e e ge

☐	6055494	April 2000	Friedman	704/9
☐	6076088	June 2000	Paik et al.	707/5

OTHER PUBLICATIONS

Friedman, et al., "Natural Language Processing in an Operational Clinical Information System", Natural Language Engingeering, vol. 1(1), pp. 83-108, May 1995.
G. Hripcsak, C. Friedman, P. Alderson, W. DuMouchel, S. Johnson, P. Clayton, "Unlocking Clinical Data From Narrative Reports," Ann. of Int. Med., vol. 122(9), pp. 681-688 (1995).
P. Haug, D. Ranum, P. Frederick, "Computerized Extraction of Coded Findings from Free-Text Radiologic Report," Radiology, vol. 174, pp. 543-548 (1990).
P. Zweigenbaum, B. Bachimont, J. Bouaud, J. Charlet and J. A. Boisvieux, "A Multi-lingual Architecture for Building a Normalized Conceptual Representation from Medical Language," Proceedings of the 19th Annual SCAMC; pp. 357-361 (1995).
R. Baud, A. Rassinoux, J. Scherrer, "Natural Language Processing and Semantical Representation of Medical Texts," Meth. of Info. Med., vol. 31(2), pp. 117-125 (1993).
L. Lenert and M. Tovar, "Automated Linkage of Free-Text Descriptions of Patients with a Practice Guideline," Proceedings of the 17th Annual SCAMC, pp. 274-278 (Ozbolt ed. 1993).
M. Gundersen, P. Haug, T. Pryor, R. van Bree, S. Koehler, K. Bauer, B. Clemons, "Development and Evaluation of a Computerized Admission Diagnoses Encoding System," Computers and Biomedical Research, vol. 29, pp. 351-372 (1996).
C. Lovis, J. Gaspoz, R. Baud, P. Michel and J. Scherrer, "Natural Language Processing and Clinical Support to Improve the Quality of Reimbursement Claim Databases," Proceedings of the 1996 AMIA Fall Annual Symposium, p. 899 (Henley & Belfus 1996).
N. Sager et al., "Medical Language Processing With SGML Display," Proceedings of the 1996 AMIA Fall Annual Symposium, pp. 547-551 (1996).
P. Zweigenbaum et al., "From Text to Knowledge: A Unifying Document-Oriented View of Analyzed Medical Language," Workshop on Medical Concept Representation and Natural Language Processing, IMIA WG6, pp. 21-29 (1997).
R. Dolin et al., "SGML as a Message Interchange Format in Healthcare," Proceeding of the 1997 Fall AMIA Annual Symposium, pp. 635-639 (1997).

ART-UNIT: 277

PRIMARY-EXAMINER: Thomas; Joseph

ATTY-AGENT-FIRM: Baker Botts LLP

ABSTRACT:

A computerized method for extracting information from natural-language text data includes parsing the text data to determine the grammatical structure of the text data and regularizing the parsed text data to form structured word terms. The parsing step, which can be performed in one or more parsing modes, includes the step of referring to a domain parameter having a value indicative of a domain from which the text data originated, wherein the domain parameter corresponds to one or more rules of grammar within a knowledge base related to the domain to be applied for parsing the text data. Preferably, the structured output is mapped back to the words in the original sentences of the text data input using XML tags.

39 Claims, 9 Drawing figures

h e b b g e e e f c e e ge

9|757227

First Hit Fwd Refs Previous Doc Next Doc Go to Doc#

Generate Collection Print

L1: Entry 1 of 3 File: USPT Jan 28, 2003

US-PAT-NO: 6513059
DOCUMENT-IDENTIFIER: US 6513059 B1

TITLE: Adaptive collaborative intelligent network system

DATE-ISSUED: January 28, 2003

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Gupta; Pradeep	Milpitas	CA		
Kondratiev; Dmitri	Moscow			RU

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
Cambira Corporation	Santa Clara	CA			02

APPL-NO: 09/ 648299 [PALM]
DATE FILED: August 24, 2000

INT-CL: [07] G06 F 15/16

US-CL-ISSUED: 709/202
US-CL-CURRENT: 709/202

FIELD-OF-SEARCH: 709/202, 709/203, 709/249

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

Search Selected Search ALL Clear

	PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
☐	6134548	October 2000	Gottsman et al.	707/5
☐	6199099	March 2001	Gershman et al.	709/203
☐	6275859	August 2001	Weseley et al.	709/229
☐	6356905	March 2002	Gershman	707/10
☐	6401085	June 2002	Gershman et al.	707/4

ART-UNIT: 2155

h e b b g e e e f c e eg e ge

PRIMARY-EXAMINER: Eng; David Y.

ATTY-AGENT-FIRM: Schipper; John F.

ABSTRACT:

System and method for facilitating exchange of information on a computer network, such as the Internet. The system provides one or more context trees, with each tree including two or more connected nodes, each node being associated with one or more selected node objects. Associated with each node is a blackboard for receiving and making available for reading, messages concerning the node object, a knowledge base containing information, facts, constraints and-or rules (Rules) concerning the node object, and an inference engine providing at least one logical rule that can be used to infer a logical conclusion based on at least one Rule in the knowledge base. A tree has a collection of at least two mobile intelligent agents that are configured to facilitate exchange of information on a node object, between two agents or between a node and an agent. An agent may migrate from a first node to a second node connected to the first node. An agent, by subscribing to the Rules of a node, is permitted to post a message on, and to read a message posted on, a blackboard for the subscribed node. The collection of agents has at least one tree agent that has knowledge of nodes that are directlz connected to each node in the tree.

34 Claims, 8 Drawing figures

Previous Doc Next Doc Go to Doc#

First Hit Fwd Refs Previous Doc Next Doc Go to Doc#
End of Result Set

Generate Collection Print

L1: Entry 3 of 3 File: USPT Jan 30, 2001

US-PAT-NO: 6182029
DOCUMENT-IDENTIFIER: US 6182029 B1

TITLE: System and method for language extraction and encoding utilizing the parsing of text data in accordance with domain parameters

DATE-ISSUED: January 30, 2001

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Friedman; Carol	Larchmont	NY		

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
The Trustees of Columbia University in the City of New York	New York	NY			02

APPL-NO: 09/ 370329 [PALM]
DATE FILED: August 6, 1999

PARENT-CASE:
SPECIFICATION This application is a continuation-in-part of U.S. application Ser. No. 08/738,889 filed Oct. 28, 1996, now U.S. Pat. No. 6,055,494 which is incorporated herein by reference.

INT-CL: [07] G06 F 17/27

US-CL-ISSUED: 704/9; 707/513, 707/531
US-CL-CURRENT: 704/9; 715/513, 715/531

FIELD-OF-SEARCH: 704/9, 704/1, 704/10, 705/2, 705/3, 707/104, 707/513, 707/531, 707/532, 707/533, 707/534, 706/45, 706/46, 706/55, 706/59

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

Search Selected Search ALL Clear

	PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
☐	5799268	August 1998	Boguraev	704/9
☐	5832496	November 1998	Anand et al.	707/102
	6038668	March 2000	Chipman et al.	713/201

h e b b g e e e f c e eg e ge

☐				
☐	6055494	April 2000	Friedman	704/9
☐	6076088	June 2000	Paik et al.	707/5

OTHER PUBLICATIONS

Friedman, et al., "Natural Language Processing in an Operational Clinical Information System", Natural Language Engingeering, vol. 1(1), pp. 83-108, May 1995.
G. Hripcsak, C. Friedman, P. Alderson, W. DuMouchel, S. Johnson, P. Clayton, "Unlocking Clinical Data From Narrative Reports," Ann. of Int. Med., vol. 122(9), pp. 681-688 (1995).
P. Haug, D. Ranum, P. Frederick, "Computerized Extraction of Coded Findings from Free-Text Radiologic Report," Radiology, vol. 174, pp. 543-548 (1990).
P. Zweigenbaum, B. Bachimont, J. Bouaud, J. Charlet and J. A. Boisvieux, "A Multi-lingual Architecture for Building a Normalized Conceptual Representation from Medical Language," Proceedings of the 19th Annual SCAMC; pp. 357-361 (1995).
R. Baud, A. Rassinoux, J. Scherrer, "Natural Language Processing and Semantical Representation of Medical Texts," Meth. of Info. Med., vol. 31(2), pp. 117-125 (1993).
L. Lenert and M. Tovar, "Automated Linkage of Free-Text Descriptions of Patients with a Practice Guideline," Proceedings of the 17th Annual SCAMC, pp. 274-278 (Ozbolt ed. 1993).
M. Gundersen, P. Haug, T. Pryor, R. van Bree, S. Koehler, K. Bauer, B. Clemons, "Development and Evaluation of a Computerized Admission Diagnoses Encoding System," Computers and Biomedical Research, vol. 29, pp. 351-372 (1996).
C. Lovis, J. Gaspoz, R. Baud, P. Michel and J. Scherrer, "Natural Language Processing and Clinical Support to Improve the Quality of Reimbursement Claim Databases," Proceedings of the 1996 AMIA Fall Annual Symposium, p. 899 (Henley & Belfus 1996).
N. Sager et al., "Medical Language Processing With SGML Display," Proceedings of the 1996 AMIA Fall Annual Symposium, pp. 547-551 (1996).
P. Zweigenbaum et al., "From Text to Knowledge: A Unifying Document-Oriented View of Analyzed Medical Language," Workshop on Medical Concept Representation and Natural Language Processing, IMIA WG6, pp. 21-29 (1997).
R. Dolin et al., "SGML as a Message Interchange Format in Healthcare," Proceeding of the 1997 Fall AMIA Annual Symposium, pp. 635-639 (1997).

ART-UNIT: 277

PRIMARY-EXAMINER: Thomas; Joseph

ATTY-AGENT-FIRM: Baker Botts LLP

ABSTRACT:

A computerized method for extracting information from natural-language text data includes parsing the text data to determine the grammatical structure of the text data and regularizing the parsed text data to form structured word terms. The parsing step, which can be performed in one or more parsing modes, includes the step of referring to a domain parameter having a value indicative of a domain from which the text data originated, wherein the domain parameter corresponds to one or more rules of grammar within a knowledge base related to the domain to be applied for parsing the text data. Preferably, the structured output is mapped back to the words in the original sentences of the text data input using XML tags.

39 Claims, 9 Drawing figures

h e b b g e e e f c e eg e ge

Previous Doc Next Doc Go to Doc#

h e b b g e e e f c e eg e ge

L1: Entry 2 of 3 File: USPT Oct 30, 2001

US-PAT-NO: 6311327
DOCUMENT-IDENTIFIER: US 6311327 B1

TITLE: Method and apparatus for analyzing software in a language-independent manner

DATE-ISSUED: October 30, 2001

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
O'Brien; Stephen Caine	Tigard	OR		
Maxwell, III; Sidney R.	Bothell	WA		

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
Applied Microsystems Corp.	Redmond	WA			02

APPL-NO: 09/ 250126 [PALM]
DATE FILED: February 12, 1999

PARENT-CASE:
CROSS-REFERENCE TO RELATED APPLICATION This application is a continuation-in-part of U.S. patent application Ser. No. 09/035,308, filed Mar. 2, 1998, now U.S. Pat. No. 6,161,200.

INT-CL: [07] G06 F 9/44

US-CL-ISSUED: 717/4; 717/8, 714/35, 714/38, 714/45
US-CL-CURRENT: 717/114; 714/35, 714/38, 714/45, 717/128, 717/130, 717/131, 717/140

FIELD-OF-SEARCH: 717/4, 717/8, 714/35, 714/38, 714/39, 714/45, 712/227

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

	PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
☐	5265254	November 1993	Blasciak et al.	717/4
☐	5408650	April 1995	Arsenault	395/575
☐	5450586	September 1995	Kuzara et al.	717/4
☐	5581695	December 1996	Knoke et al.	714/28

h e b b g e e e f c e eg e ge

☐	5615332	March 1997	Yamamoto	714/38
☐	5737520	April 1998	Gronlund et al.	714/39
☐	5748878	May 1998	Rees et al.	714/38
☐	5771345	June 1998	Tallman et al.	714/30
☐	5903759	May 1999	Sun et al.	717/4
☐	5954826	September 1999	Herman et al.	714/46
☐	6016557	January 2000	Kasprzyk et al.	714/38
☐	6047390	April 2000	Butt et al.	714/43
☐	6094730	July 2000	Lopez et al.	714/28
☐	6106571	August 2000	Maxwell	717/4
☐	6161200	December 2000	Rees et al.	714/38

FOREIGN PATENT DOCUMENTS

FOREIGN-PAT-NO	PUBN-DATE	COUNTRY	US-CL
0 567 722 A2	November 1993	EP	

OTHER PUBLICATIONS

Crooks, Roger "Embedded RISC .mu.Ps Present New Debugging Challenges," EDN, 39 (16):105-112, Aug. 4, 1994.

ART-UNIT: 212

PRIMARY-EXAMINER: Dam; Tuan Q.

ATTY-AGENT-FIRM: Seed IP Law Group PLLC

ABSTRACT:

A software analysis system for capturing tags generated by tag statements in instrumented source code. The software analysis system includes a probe that monitors the address and data bus of the target system. When a tag statement is executed in the target system, a tag is written to a predetermined location in the address space of the target. The tag contains a tag value that is indicative of the location in the source code of the tag statement generating the tag. By monitoring the predetermined address, the probe is able to capture tags as they are written on the data bus of the target system. The source code instrumenter includes a language-dependent parser and a language-independent analyzer that records tagging data in a symbol database. The software analysis system may reference the tagging data in the symbol database while testing instrumented source code. The software analysis system performs a variety of analysis functions in essentially real time, including code coverage, function and task execution times, memory allocation, call pairs, and program tracing.

34 Claims, 19 Drawing figures

Previous Doc Next Doc Go to Doc#

h e b b g e e e f c e eg e ge